Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации
Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский технический университет связи и информатики»

Кафедра Математической кибернетики и информационных технологий

Отчет по

по дисциплине «Введение в ИТ»

Выполнил: студент группы БВТ903

Нкурикийе Хафидати

Руководитель:

Мосева Марина Сергеевна

Москва 2021

1. Переменные res – это значения val или настоящие переменные var?
Ответ: val

2. "crazy" * 3 в REPL

```
scala> "crazy" * 3
res0: String = crazycrazycrazy
```

3. Что означает выражение 10 max 2? В каком классе определен метод max?

```
scala> 10 max 2
res1: Int = 10
```

Означает максимальное из двух чисел (в данном случае 10), метод определен в RichInt.

4. Используя число типа BigInt, вычислите 2^1024

```
scala> BigInt(2) pow 1024
res2: scala.math.BigInt = 179769313486231590772930519078902473361797697894230657273
430081157732675805500963132708477322407536021120113879871393357658789768814416622249
284743063947412437776789342486548527630221960124609411945308295208500576883815068234
246288147391311054082723716335051068458629823994724593847971630483535632962422413 7
216
```

5. Что нужно импортировать, чтобы найти случайное простое число вызовом метода probablePrime(100, Random) без использования каких-либо префиксов перед именами probablePrime и Random

Нужно импортировать scala.util.Random, scala.math.BigInt, scala.math.BigInt.probableInt

6. Один из способов создать файл или каталог со случайным именем состоит в том, чтобы сгенерировать случайное число типа BigInt и преобразовать его в систему счисления по основанию 36, в результате

получится строка, такая как "qsnvbevtomcj38o06kul". Отыщите в Scaladoc методы, которые можно было бы использовать для этого.

```
scala> probablePrime(100,Random) toString 36
res6: String = 28oac9mn96x6weda18dn
```

7. Как получить первый символ строки в языке Scala? А последний символ?

Чтобы получить первый символ надо использовать метод head, чтобы получить последний – last. Например:

```
scala> "My head". head
res7: Char = M

scala> "My last".last
res8: Char = t
```

8. Что делают строковые функции take, drop, takeRight и dropRight?

Take - выделяет подстроку с начала строки.
TakeRight выделятт подстроку с конца строки.
Drop - выделяет подстроку, удаляя символы в начале строки
DropRight выделяет подстроку, удаляя символы в конце строки.
Команды более простые и понятные, но нельзя, как с substring выделить подстроку из середины строки.

9. Сигнум числа равен 1, если число положительное. -1 – если отрицательное, и 0 – если равно нулю. Напишите функцию, вычисляющую это значение.

```
scala> def singnum(num:Int) = if (num>0) 1 else if(num<0) -1 else 0
singnum: (num: Int)Int

scala> singnum(6586458)
res13: Int = 1

scala> singnum(-78645)
res14: Int = -1

scala> singnum(0)
res15: Int = 0
```

10 Какое значение возвращает блок {}? Каков его тип?

Ответ: Unit.

```
scala> var empty={}
empty: Unit = ()

scala> empty
```

11. Напишите на языке Scala цикл, эквивалентный циклу на языке Java for (int i=10; i>=0; i--) System.out.println(i)

```
scala> for (i<- 10 to 0 by -1) println(i)
10
9
8
7
6
5
4
3
2
1
0
```

12. Напишите процедуру countdown (n: Int), которая выводит числа от n до 0

```
scala> def countdown(n:Int)=for(i<-n to 0 by -1) println(i)
countdown: (n: Int)Unit

scala> countdown(5)
5
4
3
2
1
0

scala>
```

13. Напишите цикл for для вычисления кодовых пунктов Юникода всех букв в строке. Например, произведение символов в строке «Hello» равно 9415087488L.

```
scala> def mySum(str:String):Long={
     | var res:Long=1
     | for(i<-str) res = res * i.toLong
     | res
     | }
mySum: (str: String)Long

scala> mySum("Hello")
res19: Long = 9415087488
```

14 Решите предыдущее упражнение без применения цикла. Напишите функцию product(s: String), вычисляющую произведение, как описано в предыдущих упражнениях.

```
scala> def func15(s:String):Long={
     | if(s.length==1) return s.charAt(0).toLong
     | else s.take(1).charAt(0).toLong *func15(s.drop(1))}
func15: (s: String)Long

scala> func15("Hello")
res63: Long = 9415087488

scala>
```

15 Сделайте функцию из предыдущего упражнения рекурсивной.

```
scala> def func15(s:String):Long={
     | if(s.length==1) return s.charAt(0).toLong
     | else s.take(1).charAt(0).toLong *func15(s.drop(1))}
func15: (s: String)Long

scala> func15("Hello")
res63: Long = 9415087488

scala>
```

16. Напишите функцию, вычисляющую xn, где n – целое число. Используйте следующее рекурсивное определение:

a. xn = y2, если n – четное и положительное число, где y=xn/2

b. xn = x*xn-1, если n – нечетное и положительное число.

c. x0=1.

d. xn=1/x-n, если n – отрицательное число. Не используйте инструкцию return.

```
scala>  def pow(x:Int, n:Int)=n match{
     |  case n if n>0 && (n%2 ==0) =>pow(x,n/2)* pow(x,n/2)
     |  case n if n>0 && (n%2==1)=> x* pow(x,n-1)
     | case n if n<0 => 1/pow(x,-n)
     | case 0 => 1
     | }
pow: (x: Int, n: Int)Double
```

```
scala> pow(4,2)
res21: Double = 16.0
```

18. f(m,n) - сумма всех натуральных чисел от m до n включительно, в десятичной записи которых нет одинаковых цифр.

```
scala> def unicCh(n:Int):Boolean={
     | var list1:List[Int]=List()
     | var num:Int=n
     | while(num>0){
     | var ch:Int=num%10
     | if(list1.contains(ch))return false
     | else list1=list1:+ch
     | num=num/10
     | }
     | return true}
unicCh: (n: Int)Boolean

scala> unicCh(1233)
res59: Boolean = false

scala> unicCh(123)
res60: Boolean = true

scala> def func18(m:Int,n:Int):Int={
     | var sum:Int=0
     | for(i<-m to n){
     | if(unicCh(i)) sum+=i}
     | sum}
func18: (m: Int, n: Int)Int

scala> func18(1,10)
res61: Int = 55
```

19. Список содержит целые числа, а также другие списки, такие же как и первоначальный. Получить список, содержащий только целые числа из всех вложенных списков.

Пример: f(List(List(1, 1), 2, List(3, List(5, 8)))) = List(1, 1, 2, 3, 5, 8)

```
scala> var func19_ans: List[Any] = List()
func19_ans: List[Any] = List()

scala> def func19(n: List[Any]): List[Any]={
     |   for (i <- n){
     |     if (i.isInstanceOf[Int]) func19_ans = func19_ans :+ i
     |     else (func19(i.asInstanceOf[List[Any]]))
     |   }
     |   func19_ans
     | }
func19: (n: List[Any])List[Any]

scala> func19(List(List(1,1),2,List(3,List(5,8))))
res5: List[Any] = List(1, 1, 2, 3, 5, 8)
```

20. f(n) - сумма цифр наибольшего делителя натурального числа n.

```
scala> def bigDelitel(n:Int):Int={
     | for(i<-1 to n-1 reverse){
     | if(n%i==0) return i}
     | -1}
warning: one feature warning; for details, enable `:setting -feature' or `:replay -feature'
bigDelitel: (n: Int)Int

scala> def f20(n:Int):Int={
     | var res =0;
     | var bigDel=bigDelitel(n)
     | for(i<-bigDel.toString) res+=i.asDigit
     | res
     | }
f20: (n: Int)Int
```

21. Список содержит элементы одного, но любого типа. Получить список, содержащий каждый имеющийся элемент старого списка k раз подряд. Число k задается при выполнении программы.

```
scala> def myfunc21(n:List[Any],k:Int){
     | var res:List[Any]=List()
     | for(i<-n){
     | for(j<-1 to k){
     | res=res:+i}
     | }
     | println(res)
     | }
myfunc21: (n: List[Any], k: Int)Unit
```

24. f(m,n) - наименьшее общее кратное натуральных чисел m и n

```
scala> def myEvclid(num1:Int,num2:Int):Int={
     | if(num1==0 || num2==0) return num1+num2
     | else if(num1>num2) return myEvclid (num1%num2,num2)
     | else return return myEvclid (num1,num2%num1)
     | }
myEvclid: (num1: Int, num2: Int)Int

scala> def func24(num1:Int,num2:Int):Int={
     | num1*num2/myEvclid(num1,num2)}
func24: (num1: Int, num2: Int)Int

scala> func24(145,45)
res58: Int = 1305
```

25. Список содержит элементы одного, но любого типа. Получить список, из элементов исходного, удаляя каждый k-й элемент. Число k задается при выполнении программы.

```
scala> def func25(n:List[Any],del:Int){
     | var res:List[Any]=List()
     | var count:Int=0;
     | for(i<-n){
     | count+=1
     | if(count<del) res=res:+i
     | else count=0
     | }
     | println(res)}
func25: (n: List[Any], del: Int)Unit

scala> func25(List(1,2,3,4,5,6,7,8,9),2)
List(1, 3, 5, 7, 9)

scala> _
```

26. f(n,k) - число размещений из n по k. Факториал не использовать.

```
scala> def myfactorial(n:Int):Int={
     | var res:Int=1
     | for(i<-1 to n) res=res * i
     | res}
myfactorial: (n: Int)Int

scala> def myfunc26(n:Int,k:Int):Int={
     | if(k>n) -1
     | else {
     | myfactorial(n)/myfactorial(n-k)}
     | }
myfunc26: (n: Int, k: Int)Int

scala> myfunc26(10,4)
res36: Int = 5040
```

27. Список содержит элементы одного, но любого типа. Получить новый список, перемещая циклически каждый элемент на k позиций влево (при перемещении на одну позицию первый элемент становится последним, второй первым и так далее). Число k задается при выполнении программы. Если k отрицательное, то перемещение происходит вправо

```
scala>   def func27(n: List[Any], moveleft: Int){
     |     val indexlength = n.length - 1
     |     var ans: List[Any] = n
     |
     |     if (moveleft >= 0){
     |       val left = ans.take(moveleft)
     |       val right = ans.takeRight(n.length - moveleft)
     |       ans = right ::: left
     |     }
     |     else {
     |       val right = ans.takeRight(-moveleft)
     |       val left = ans.take(n.length + moveleft)
     |       ans = right ::: left
     |     }
```

```
scala> func27(List(1,4,5,6,7,7),3)
6, 7, 7, 1, 4, 5,
```

28. f(n) - наибольшее совершенное число не превосходящее n. Совершенным называется натуральное число n равное сумме своих делителей, меньших n, например 6 = 1 + 2 + 3 (f(6) = 6, f(7) = 6, ...).

```
scala> def delSum(n:Int):Boolean={
     | var res:Boolean=false
     | var sum:Int=0
     | for(i<-1 to n-1 reverse){
     | if(n%i==0)sum+=i
     | }
     | if(n==sum) res=true
     | res}
warning: one feature warning; for details, enable `:setting -feature' or `:replay -feature'
delSum: (n: Int)Boolean

scala> def myFunc28(n:Int):Int={
     | for(i<-1 to n-1 reverse){
     | if(delSum(i))return i}
     | 1}
warning: one feature warning; for details, enable `:setting -feature' or `:replay -feature'
myFunc28: (n: Int)Int

scala> myFunc28(28539)
res26: Int = 8128
```

29. Список содержит элементы одного, но любого типа. Получить два списка из элементов исходного, выбирая в первый элементы с четными индексами, а во второй с нечетными.

```
scala> def myfunc29(n:List[Any]){
     | val len=n.length - 1
     | var list1: List[Any]=List()
     | var list2: List[Any]=List()
     | for(i<-0 to len){
     | if(i%2==0)list1=list1:+n(i)
     | else list2=list2:+n(i)
     | }
     | println(list1)
     | println(list2)
     | }
myfunc29: (n: List[Any])Unit

scala> myfunc29(List(1,2,3,4,5,6,7,8))
List(1, 3, 5, 7)
List(2, 4, 6, 8)
```

30. f(n) - наибольшее из чисел от 1 до n включительно, обладающее свойством: сумма цифр n в некоторой степени > 1 равна самому числу n. Пример: 512 = 83

```
scala> def sumChifr(n:Int):Int={
     | var sum:Int=0
     | var nn:Int=n
     | while(nn>0){
     | sum+=nn%10
     | nn=nn/10
     | }
     | sum}
sumChifr: (n: Int)Int
```

```
scala> def issPow(n:Int):Boolean={
     | var num:Int=sumChifr(n)
     | var ch:Int=num
     | if(ch==n) return true
     | while(ch<n){
     | ch*=num
     | if(ch==n) return true
     | }
     | false
     | }
issPow: (n: Int)Boolean
```

```
scala> def func30(n:Int):Int={
     | for(i<-1 to n reverse){
     | if(issPow(i)) return i}
     | -1}
warning: one feature warning; for details, enable `:setting -feature' or `:replay -feature'
func30: (n: Int)Int

scala> func30(515)
res49: Int = 512
```

31. Список в качестве элементов содержит кортежи типа: (n, s), где n — целые числа, а s — строки. Получить два списка из элементов исходного, выбирая в первый числа, а во второй строки из кортежей.

```
scala> def myfunc31(n:List[Any]){
     | val len=n.length -1
     | var list1,list2:List[Any]=List()
     | for(i<-0 to len){
     | list1=list1:+n(i).asInstanceOf[List[Any]](0)
     | list2=list2:+n(i).asInstanceOf[List[Any]](1)
     | }
     | println(list1)
     | println(list2)
     | }
myfunc31: (n: List[Any])Unit

scala> myfunc31(List(List(12,"hello"),List(13,"world")))
List(12, 13)
List(hello, world)
```